PvMail Documentation

Release 3 (3.0-r675)

Pete Jemian, APS, ANL <jemian@anl.gov>

November 29, 2011

CONTENTS

https://subversion.xor.aps.anl.gov/bcdaext/pvMail/doc/build/html/index.html

PvMail was built to watch (monitor) an EPICS PV and send an email when the value of that PV changes from 0 to 1.

The PV being watched (that *triggers* the sending of the email) can be any EPICS record type or field that results in a value of 0 (zero) that changes to 1 (one). This includes these EPICS records (and possibly more): *ai*, *ao*, *bi*, *bo*, *calcout*, *scalcout*, *swait*, ...

When an event causes an email to be triggered, PvMail will retrieve the value of another PV that is the first part of the message to be sent. Additional metadata will be appended to the message.

Note: Email is sent using a call to the `mail` program on the native OS. This almost certainly precludes its use on Windows systems. The GUI or command-line versions will operate but likely no email is sent. Also, the host computer must allow sending email to the intended recipients.

PvMail provides either a command-line interface or a graphical user interface. Both are accessed from the same command, using different command-line options. The command-line version is intended to run as a background program, it has no user interaction but logs all its output into a log file. The GUI version provides a screen to edit each of the parameters before the background process is started. It also provides buttons to start and stop the background process.

CHAPTER

ONE

OVERVIEW OF CONTENTS

1.1 Contents

1.1.1 pvMail.py: command-line interface

Basically, you use it either as a background daemon or as a GUI. Call it with a `-g` or `--gui` command line option to force the GUI to run, otherwise you get the background daemon. Either way, it makes a log file (based on PID number) with any program output.

background daemon:

```
pvMail.py triggerPV messagePV user1@email.domain,user2@host.server &
```

GUI:

```
pvMail.py triggerPV messagePV user1@email.domain,user2@host.server -g &
```

PvMail uses Matt Newville's PyEpics package for EPICS CA connections and Enthought's Traits package to build the GUI (which means it could use either WX or QT).

Starting PvMail from the command-line

PvMail is started from the command line:

```
$ ./pvMail.py pvMail:trigger pvMail:message jemian
```

No program output is printed to the screen. Instead, the output is directed to a log file. Here is an example:

```
INFO:root:(pvMail.py,2011-11-27 19:03:23.072392) ####################################################
INFO:root:(pvMail.py,2011-11-27 19:03:23.072826) startup
INFO:root:(pvMail.py,2011-11-27 19:03:23.072897) trigger PV       = pvMail:trigger
INFO:root:(pvMail.py,2011-11-27 19:03:23.073323) message PV       = pvMail:message
INFO:root:(pvMail.py,2011-11-27 19:03:23.073401) email list       = ['jemian']
INFO:root:(pvMail.py,2011-11-27 19:03:23.073463) log file         = logfile.log
INFO:root:(pvMail.py,2011-11-27 19:03:23.073553) message file     = pvmail_email.txt
INFO:root:(pvMail.py,2011-11-27 19:03:23.073667) logging interval = 5.0
INFO:root:(pvMail.py,2011-11-27 19:03:23.073735) sleep duration   = 0.2
INFO:root:(pvMail.py,2011-11-27 19:03:23.073795) interface        = command-line
INFO:root:(pvMail.py,2011-11-27 19:03:23.073855) user             = jemian
INFO:root:(pvMail.py,2011-11-27 19:03:23.073952) host             = como-ubuntu64
INFO:root:(pvMail.py,2011-11-27 19:03:23.074053) program          = ./pvMail.py
INFO:root:(pvMail.py,2011-11-27 19:03:23.074124) PID              = 8903
INFO:root:(pvMail.py,2011-11-27 19:03:23.074196) do_start
```

```
INFO:root:(pvMail.py,2011-11-27 19:03:23.074280) test connect with pvMail:message
INFO:root:(pvMail.py,2011-11-27 19:03:23.445334) test connect with pvMail:trigger
INFO:root:(pvMail.py,2011-11-27 19:03:23.468540) passed basicChecks(), starting monitors
INFO:root:(pvMail.py,2011-11-27 19:03:23.477917) checkpoint
INFO:root:(pvMail.py,2011-11-27 19:03:27.373142) pvMail:trigger = 1
INFO:root:(pvMail.py,2011-11-27 19:03:27.373908) SendMessage
INFO:root:(pvMail.py,2011-11-27 19:03:27.374199) sending email to: jemian
INFO:root:(pvMail.py,2011-11-27 19:03:27.374716) mail -s "pvMail.py: pvMail:trigger" jemian < /tmp/pv
INFO:root:(pvMail.py,2011-11-27 19:03:27.538022) message(s) sent
INFO:root:(pvMail.py,2011-11-27 19:03:28.092551) checkpoint
INFO:root:(pvMail.py,2011-11-27 19:03:29.440516) pvMail:trigger = 0
```

The program starts, reports its configurations, and connects with the EPICS PVs, and then goes into a background mode. A checkpoint (command-line option `-i`) is reported perdiocally. The default is 5 seconds. This may be changed to 10 minutes or longer for production use, but is always specified in seconds.

Observe that, in the above example, the trigger PV changed from 0 to 1 at 19:03:27.373142 (and back to 0 at 19:03:29.440516). The change at ~19:03:27 triggered PvMail to send an email as configured. For now, the code writes the text of the email to a temporary file (command-line option `-m`, default is "/tmp/pvmail_message.txt"). In this example, the message reads:

```
pvMail default message

user: jemian
host: como-ubuntu64
date: 2011-11-27 19:03:27.374135
program: ./pvMail.py
PID: 8903
trigger PV: pvMail:trigger
message PV: pvMail:message
recipients: jemian
```

The message shows up in the mail browser (here my Linux `mail` program):

```
jemian@como-ubuntu64$ mail
Mail version 8.1.2 01/15/2001.  Type ? for help.
"/var/mail/jemian": 3 messages 3 new
>N  1 jemian@como-ubunt  Sun Nov 27 18:27   25/730   pvMail.py: pvMail:trigger
 N  2 jemian@como-ubunt  Sun Nov 27 18:58   25/730   pvMail.py: pvMail:trigger
 N  3 jemian@como-ubunt  Sun Nov 27 19:03   25/730   pvMail.py: pvMail:trigger
```

The full message, as seen in the mail browser is:

```
Message 3:
From jemian@como-ubuntu64 Sun Nov 27 19:03:27 2011
Envelope-to: jemian@como-ubuntu64
Delivery-date: Sun, 27 Nov 2011 19:03:27 -0600
To: jemian@como-ubuntu64
Subject: pvMail.py: pvMail:trigger
From: Pete R Jemian <jemian@como-ubuntu64>
Date: Sun, 27 Nov 2011 19:03:27 -0600

pvMail default message

user: jemian
host: como-ubuntu64
date: 2011-11-27 19:03:27.374135
program: ./pvMail.py
PID: 8903
```

```
trigger PV: pvMail:trigger
message PV: pvMail:message
recipients: jemian
```

Starting PvMail from the command-line at the APS

At the APS, Enthought Python Distribution is installed on the /APSshare partition available to all beam lines.

Here is a command to run the PvMail and get the help message:

```
/APSshare/epd/rh5-x86_64/bin/python /APSshare/epd/demos/pvMail.py -h
```

or the 32-bit version:

```
/APSshare/epd/rh5-x86/bin/python /APSshare/epd/demos/pvMail.py -h
```

Note: Support for both RHEL5 and RHEL6 use the same Enthought Python Distribution.

1.1.2 command-line parameters

usage

When PvMail is started from the command line with no additional parameters:

```
$ pvMail.py

usage: pvMail.py [-h] [-l LOG_FILE] [-m MESSAGE_FILE] [-i LOGGING_INTERVAL]
                 [-r SLEEP_DURATION] [-g] [-v]
                 trigger_PV message_PV email_addresses
pvMail.py: error: too few arguments
```

This is the *usage* message. It tells us we must supply three positional arguments: `trigger_PV message_PV email_addresses`.

positional argument: `trigger_PV`

EPICS process variable name to watch using a CA monitor. When `trigger_PV` makes a transition from 0 (zero) to 1 (one), then get the string from the `message_PV` and send an email to all of the `email_addresses` on the list.

positional argument: `message_PV`

EPICS process variable name pointing to a (short) message that will be used as the first part of the email message to be sent.

positional argument: `email_addresses`

List of email addresses, separated by commas if more than one. For example, `user1@email.domain,user2@host.server` will send one email to `user1@email.domain` and another email to `user2@host.server`.

option: `--version` or `-v`

The current version of the program can always be printed using the `-v` or `--version`. With this option, the program prints the version number and then quits.

```
$ pvMail.py --version
3.0-663
```

option: `--help` or `-h`

It may be easier to review the short help instructions for command-line options:

```
$ ./pvMail.py --help
usage: pvMail.py [-h] [-l LOG_FILE] [-m MESSAGE_FILE] [-i LOGGING_INTERVAL]
                 [-r SLEEP_DURATION] [-g] [-v]
                 trigger_PV message_PV email_addresses

Watch an EPICS PV. Send email when it changes from 0 to 1.

positional arguments:
  trigger_PV          EPICS trigger PV name
  message_PV          EPICS message PV name
  email_addresses     email address(es), comma-separated if more than one

optional arguments:
  -h, --help          show this help message and exit
  -l LOG_FILE         for logging program progress and comments
  -m MESSAGE_FILE     temporary file for email message
  -i LOGGING_INTERVAL checkpoint reporting interval (s) in log file
  -r SLEEP_DURATION   sleep duration (s) in main event loop
  -g, --gui           Use the graphical rather than command-line interface
  -v, --version       show program's version number and exit
```

option: `--gui` or `-g`

This command line option is used to start the GUI (see *pvMail.py: graphical user interface*). If either GUI option is used, then the positional arguments (`triggerPV messagePV email@address`) are optional.

option: `-l LOG_FILE`

Both the command-line and GUI versions of PvMail log all program output to a log file. If a LOG_FILE is not specified on the command line, the default file will be `pvMail-PID.log` in the current directory where *PID* is the process identifier of the running pvMail.py program.

Note: If the LOG_FILE already exists, new information will be appended. It is up to the account owner to delete a LOG_FILE when it is no longer useful.

The PID number is useful when you wish to end a program that is running as a background daemon. The UNIX/Linux command is:

```
kill PID
```

option: `-m MESSAGE_FILE`

To send an email message, PvMail writes the content of the message to a temporary file. If MESSAGE_FILE is not specified on the command line, the default file will be `pvmail_email.txt` in the current directory.

This file will be overwritten each time a new message is to be sent.

option: `-i LOGGING_INTERVAL`

units seconds

When a program runs in the background, waiting for occasional activity, there is often some concern that the program is actually prepared to act when needed. To offset this concern, PvMail will report a *checkpoint* message periodically (every LOGGING_INTERVAL seconds, default is every 5 minutes) to the LOG_FILE. The program ensures that LOGGING_INTERVAL is no shorter than 5 seconds or longer than 1 hour.

option: `-r SLEEP_DURATION`

units seconds

For operation as a background daemon process, the command-line version must check periodically for new EPICS CA events, using a call to `epics.ca.poll()`. In between calls, the application is told to sleep for SLEEP_DURATION seconds. The default SLEEP_DURATION is 0.2 seconds and is limited to values between 0.1 ms and 5 s.

1.1.3 pvMail.py: graphical user interface

The PvMail program GUI is started from the command line with the `-g` or `--gui` command-line options. If either GUI option is used, then the positional arguments (`triggerPV messagePV email@address`) are optional. Without either GUI option, the command-line interface is started.:

```
$ ./pvMail.py -g &
```

Tip: Usually, you want to run the GUI as a background task by appending the ampersand (`&`) on the command line, as shown above.

The GUI provides editable text entry widgets for each of the required command-line terms (a.k.a. *positional arguments*): `trigger_PV message_PV email_addresses`. The list of email addresses is separated. The GUI provides a tool to add additional address or remove addresses.

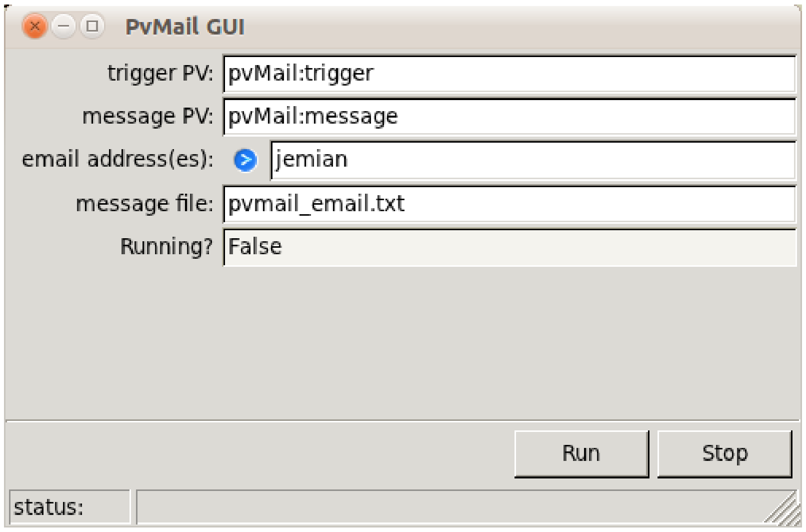
Additionally, the GUI provides an entry box to change the name of the MESSAGE_FILE (a temporary file used to store the content of the email message).

Warning: At present, the GUI provides few visual cues about the success of PV connections or even that the program is working. This will be fixed soon.

The GUI also shows (using *True* or *False* text) whether or not the PV monitor process is running.

Warning: If either of the PVs fails to connect, it is likely that the GUI may become confused whether or not it is running. In such cases, press the *Stop* button, then press the *Run* button to try to restart monitoring.

All PvMail monitoring will be stopped if the GUI window is closed. At present, there is no feature to detach or reattach a monitor set. Also, PvMail can only monitor a single set of PVs using the current design. A request to enhance this capability is on the TODO list (see *TODO items for the next release*).



trigger PV: pvMail:trigger
message PV: pvMail:message
email address(es): jemian
message file: pvmail_email.txt
Running? False

Figure 1.1: GUI of the *PvMail* application

At the bottom of the GUI panel, a status line is shown. At present, this does not indicate the status of the monitoring process. Again, this will be fixed soon.

1.1.4 EPICS test database

To test the program during its development, a test database (*test.db*) was prepared. The database creates two PVs:

`pvMail:trigger` the PV to watch

`pvMail:message` the message to be sent

starting: softIoc

Start the database by adding it to an existing EPIC IOC configuration or by starting a soft IOC using the `softIoc` program softIOC from EPICS base. Here is an example of how that looks from a Linux command shell:

```
$ softIoc -d test.db
Starting iocInit
############################################################################
## EPICS R3.14.12 $Date: Wed 2010-11-24 14:50:38 -0600$
## EPICS Base built Feb 27 2011
############################################################################
iocRun: All initialization complete
epics>
```

Note: Here, the shell prompt is signified by the `$` symbol.

watching: camonitor

Once the EPICS IOC is started and the PVs are available, it is possible to watch them for any changes from the command line using the `camonitor` camonitor application from EPICS base:

```
$ camonitor pvMail:trigger pvMail:message
    pvMail:trigger                 <undefined> off UDF INVALID
    pvMail:message                 <undefined> pvMail default message UDF INVALID
```

Note: Do not be concerned about the `UDF INVALID` notices, they will disappear once the PVs have been written to at least once.

changing a PV: caput

You can test changing the value of the trigger PV using the `caput` caput application from EPICS base:

```
$ caput pvMail:trigger 1
    Old : pvMail:trigger                 off
    New : pvMail:trigger                 on
```

test.db

Here is the full listing of the test EPICS database used for program development.

```
########### SVN repository information ###################
# $Date: 2011-11-27 23:22:22 -0600 (Sun, 27 Nov 2011) $
# $Author: jemian $
# $Revision: 667 $
# $URL: https://subversion.xor.aps.anl.gov/bcdaext/pvMail/src/PvMail/test.db $
# $Id: test.db 667 2011-11-28 05:22:22Z jemian $
########### SVN repository information ###################

# EPICS database to use while testing and developing pvMail.py code

#  /APSshare/epics/base-3.14.12.1/bin/linux-x86-el5-debug/softIoc -d test.db
#
#  IOC:     softIoc -d test.db
#  client:  camonitor pvMail:{trigger,message}
#  pvMail:  pvMail.py  pvMail:trigger pvMail:message prjemian@gmail.com,jemian@anl.gov

record(bo, "pvMail:trigger")
{
        field(DESC, "trigger PV")
        field(ZNAM, "off")
        field(ONAM, "on")
}
record(stringout, "pvMail:message")
{
        field(DESC, "message to be sent by email")
        field(VAL,  "pvMail default message")
}
```

1.1.5 PvMail as a Python package

This section provides the source code documentation. The documentation here may be of little or no use to the casual user of this software.

installation

The PvMail project can be installed as a Python package.

1. Checkout the project from subversion
2. Change into the project working directory
3. Run `setup.py install`

starter program

Once the PvMail project has been installed as a package, the PvMail application can be run using a simple Python script (included in the project tree at the top-level directory as `pvMail.py`). Here is the current version of that script.

```
#!/usr/bin/env python

'''
runs the PvMail application

Documentation:
    https://subversion.xor.aps.anl.gov/bcdaext/pvMail/doc/build/html/index.html

see the help option for immediate details about the command-line::

    pvMail.py --help

'''

from PvMail import pvMail
pvMail.main()
```

1.1.6 PvMail source code documentation

PvMail Python Package

Source code documentation for EPICS PvMail.

`pvMail` Module

pvMail: combined CLI and GUI Functionally based on pvMail UNIX shell script written in 1999.

Summary Watches an EPICS PV and sends email when it changes from 0 to 1. PV value can be either integer or float.

Note: When "running", wait for trigger PV to go from 0 to 1. When that happens, fetch mail message from message PV. Then, send that message out to each of the email addresses. The message content is prioritized for view on a small-screen device such as a pager or a PDA or smartphone.

author Kurt Goetze (original version)

author Pete Jemian (this version)

organization AES/BCDA, Advanced Photon Source, Argonne National Laboratory

license Copyright (c) 2011, UChicago Argonne, LLC

license Copyright (c) 2009, The University of Chicago, as Operator of Argonne National Laboratory.

license Copyright (c) 2009 The Regents of the University of California, as Operator of Los Alamos National Laboratory.

license This file is distributed subject to a Software License Agreement found in the file LICENSE that is included with this distribution.

note Version History:

note 05.09.07 kag Initial alpha version. Needs testing.

note 2009-12-02 prj: converted to use wxPython (no Tkinter or Pmw)

note 2011-11-23 prj: complete rewrite using PyEpics and combined GUI (Traits) and CLI

requires EPICS system (http://www.aps.anl.gov/epics) with at least two process variables (PVs) where the "Trigger PV" toggles between values of 0 and 1 and the "SendMessage PV" contains a string to send as part of the email message.

requires PyEpics (http://cars9.uchicago.edu/software/python/pyepics3/)

requires Traits (http://code.enthought.com/projects/traits/)

class `PvMail.pvMail.`**`PvMail`**

Bases: `threading.Thread`

Watch an EPICS PV (using PyEpics interface) and send an email when the PV changes from 0 to 1.

`basicChecks`()

check for valid inputs, raise exceptions as discovered, otherwise no return result

`do_restart`()

restart watching for triggers

`do_start`()

start watching for triggers

`do_stop`()

stop watching for triggers

`receiveMessageMonitor`(*value*, ***kw*)

respond to EPICS CA monitors on message PV

`receiveTriggerMonitor`(*value*, ***kw*)

respond to EPICS CA monitors on trigger PV

`send_test_message`()

sends a test message, used for development only

`testConnect`(*pvname*, *timeout=5.0*)

create PV, wait for connection, return connection state (True | False)

adapted from PyEpics __createPV() method

class `PvMail.pvMail.`**`SendMessage`**(*pvm*)

Bases: `threading.Thread`

initiate sending the message in a separate thread

`PvMail.pvMail.`**`basicMailTest`**()

simple test sending mail using the PvMail class

`PvMail.pvMail.`**`basicStartTest`**`()`
: simple test of the PvMail class

`PvMail.pvMail.`**`cli`**`(`*results*`)`
: command-line interface to the PvMail class

  Parameters **results** (*obj*) – default parameters from argparse, see main()

`PvMail.pvMail.`**`gui`**`(`*results*`)`
: graphical user interface to the PvMail class

  Parameters **results** (*obj*) – default parameters from argparse, see main()

`PvMail.pvMail.`**`logger`**`(`*message*`)`
: log a report from this class.

  Parameters **message** (*str*) – words to be logged

`PvMail.pvMail.`**`main`**`()`
: parse command-line arguments and choose which interface to use

`PvMail.pvMail.`**`sendMail`**`(`*subject*, *message*, *recipients*`)`
: send an email message

  Parameters

  - **subject** (*str*) – short text for email subject
  - **message** (*str*) – full text of email body
  - **recipients** (*[str]*) – list of email addresses to receive the message

`traits_gui` Module

pvMail: just the GUI Build the Graphical User Interface for PvMail using the Traits library from the Enthought Python Distribution.

Copyright (c) 2011, UChicago Argonne, LLC

class `PvMail.traits_gui.`**`ActionHandler`**
: Bases: `traitsui.handler.Handler`

  implements controls for PvMail GUI application

  `do_run``(`*uinfo*`)`
  : start watching the EPICS triggerPV

    Parameters **uinfo** (*obj*) – UIInfo object passed from the Action()

    Traits Handler method that responds to a Traits Action()

  `do_stop``(`*uinfo*`)`
  : stop watching the EPICS triggerPV

    Parameters **uinfo** (*obj*) – UIInfo object passed from the Action()

    Traits Handler method that responds to a Traits Action()

class `PvMail.traits_gui.`**`PvMail_GUI`**`(`*triggerPV='‘, messagePV='‘, recipients=['‘, '‘], message_file='gui.log', log_file='‘, **kwtraits*`)`
: Bases: `traits.has_traits.HasTraits`

  GUI used for pvMail, declared using Enthought's Traits module

`SetStatus`(*msg*)
put text in the status box

`actionRun` = <traitsui.menu.Action object at 0x4cac1d0>

`actionStop` = <traitsui.menu.Action object at 0x4cac230>

1.1.7 More Information

subversion repository

The PvMail project is hosted on the APS XSD subversion server. You may check out the entire project source code subversion repository and development subdirectory:

```
svn co  https://subversion.xor.aps.anl.gov/bcdaext/pvMail ./pvMail
```

project management site

You may find it easier to view the various code revisions and other aspects of the project from the project management site. Links there will direct you to the resources (documentation, source code) for the PvMail project.

https://subversion.xor.aps.anl.gov/trac/bcdaext/wiki/PvMail

Documentation

Documentation for the PvMail project, maintained using sphinx (http://sphinx.pocoo.org), can be accessed from the WWW at https://subversion.xor.aps.anl.gov/bcdaext/pvMail/doc/build/html/index.html PvMail project and also `EPUB` and `PDF` versions.

1.1.8 TODO items for the next release

1. Connect status updates from `pvMail.PvMail()` with status in the GUI
2. Report PV connection problems in an obvious way
3. GUI: display the tail end of the LOG_FILE.
4. GUI: save/restore settings from a named file
5. GUI: manage multiple pvMail.PvMail() objects (starting, stopping, detaching, ...)

1.1.9 License

CHAPTER

TWO

GLOSSARY

CA EPICS Channel Access protocol

CLI command-line interface

EPICS http://www.aps.anl.gov/epics

GUI graphical user interface

IOC EPICS Input/Output Controller, the EPICS server

message PV EPICS PV that provides the text to be sent by email, additional metadata is appended to this text

OS operating system

PV EPICS process variable

PyEpics Python package to manage connections with PVs served by an EPICS IOC

Traits Python package to simplify construction of a GUI

trigger PV EPICS PV that signals an email is to be sent

CHAPTER

THREE

DEPENDENCIES

This software was built with various standard Python packages available in Python 2.7. Additionally, this program uses:

***PyEpics* (EPICS interface)** http://cars9.uchicago.edu/software/python/pyepics3/

***Traits* (GUI library)** http://code.enthought.com/projects/traits/

Both of these are available for easy_install from the Python Package Index (http://pypi.python.org/pypi).

PYTHON MODULE INDEX

INDEX